Aperture Filter Optimization

Jakub Chłapiński and Zygmunt Ciota
Technical University of Lodz
Department of Microelectronics and Computer Science
Łódź, Poland
jchlapi@dmcs.pl, ciota@dmcs.pl

Abstract— **Aperture filters are a relatively new class of image operators, where in addition to the domain constraint, imposed by windowing, the range constraint is defined, and effectively condensing probability mass to improve filter output estimates. In this paper the automatic design methodology by means of stochastic optimization is proposed, which can generate an aperture filter to suit a specific application. Usability of presented method was verified experimentally, and the results are presented in the paper.**

Index Terms—**aperture filters, mathematical morphology, image processing**

I. Introduction

The heuristic design of digital image filters is a complex and difficult task. It requires substantial signal processing knowledge and experience from the user, and can also be very time consuming. There are many filtering tasks where proper heuristics are impossible to find, using analytical signal processing techniques, due to the very complex or subtle nature of the filtered signal. In addition, most image processing software users, such as biologist, astronomers or medical scientist, do not possess sufficient knowledge to design a filter to fit their needs. However in almost every case the user can supply several, possibly manually edited, examples of both input, and desired output signal. Therefore it would be of a great advantage, if a method existed for generating an image operator from input-output examples, which could perform the desired filtration [6].

In statistical approaches to non-linear filter design the task is to find a filter Y, given the training dataset of signal $h(t)$ to be observed and corresponding signal $g(t)$ to be estimated, so that the error measure between $Y(h)(t)$ and $g(t)$ is minimized.

The general class of image operators used in automatic design is a class of window operators, or *W-operators*. A window operator determines the value of its output (usually its center pixel) from values of neighboring pixels within a specified window.

Aperture filters [1, 2, 3] are a recently introduced class of non-linear filters used in image processing. Aperture operators or *WK-operators* are a subclass of *W-operators*, which are not only defined within a window, but also restricted to a number of gray levels. This way the search space is greatly reduced but still can be significant. For a specified window W the signal is projected into the aperture K, K = [-k, k] by placing the aperture on a specified value h_a (usually the value of the center pixel or median of the signal within window W) (Fig. 1). Signal values outside the aperture are clipped to the closest aperture boundary (gray pixels in Fig. 1).

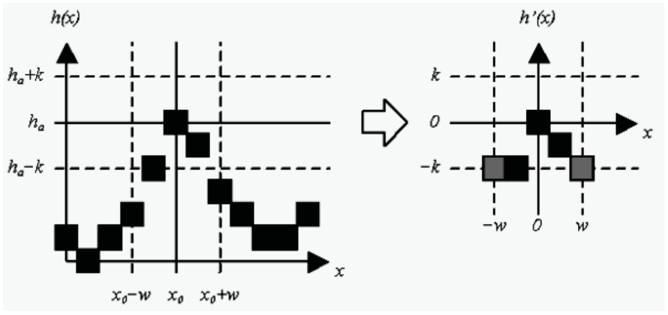

Figure 1. Example of a signal *h(x)* projected into the aperture window

The choice of window size, its shape and the number of aperture levels are the most important design considerations. In addition, a whole selection of various methods for improving filtering aperture filters quality were investigated, such as scaling, using non-rectangular mask shapes and others. The pyramidal approach for aperture filter design was proven to be especially effective. In the pyramidal approach, the filter is constructed of several aperture operators, each operating on gradually lower level-of-detail [4, 5]. In case a filter output is poorly estimated for higher level of detail, the next filter is tried, and so on, until a correctly estimated output value is found.

Example multi-resolution filter bank is presented in Fig. 2. If for a given input pattern the output estimate is not found for operator W_1, operator W_2 is tried, etc. In practical implementation of pyramidal aperture filter, flexible hierarchical filter definition is crucial in order to design and optimize such a structure. In this paper we propose universal and flexible approach for pyramidal aperture filter definition with the use of XML technology for structure description, presented in detail later.

Aperture filters can be used to for any filtering task, as long as it is possible to provide a training dataset, consisting of example image pairs with original (output) and corrupted (input) images. In real applications, obtaining such a dataset can be somewhat difficult, but often possible (for instance creating images by hand or by using artificial process, similar to the real one to be filtered). Using aperture filters is advantageous if the cost of producing the training set is lower than the cost of finding heuristic/analytical transformation for the filtering problem.

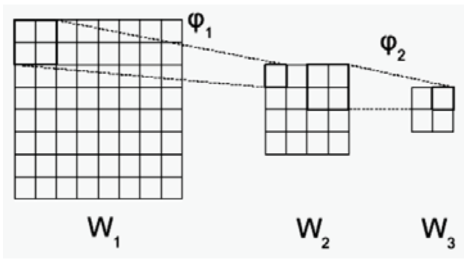

Figure 2. Example of multi-resolution filter bank

In [7] authors presented a new approach for aperture filter design, improving operator performance with respect to the MSE measure by releasing some of the operator constraints without losing statistical estimation accuracy. With the use of the proposed methods an average of 34% MSE reduction was achieved for deblurring, whereas a standard aperture operator reduced the error by only 10% on the average.

Implementing such definition of aperture filter obviously does not exhaust all the possibilities for improving filter performance. A large set of aperture filter parameters can be tuned for specific application, naturally influencing (improving or degrading) the quality of filtration.

Choosing optimal values for the parameters can be a difficult task for a human operator, as there are no straightforward means for deducing them. In practice aperture filter design takes a number of trial iterations with no clear directions of the further improvement. Usually there are a large number of adjustable parameters defining the filter structure, and a slight change in a parameter value might worsen or improve filtering quality without clear correlation from the user perspective. In effect, there are no well defined methods for aperture filter design for a given task.

It would seem that the most sensible solution for finding optimal aperture filter parameter values is to define the fitness function, which allows comparing and qualifying the performance of aperture filters with different parameters. Parameter values can than be found by optimising such function with the use of well known optimisation (minimisation) frameworks.

Non-linear nature of filtration process, as well as large complexity of aperture filter's structure, make it intractable to use analytical optimisation, or exhaustive search for finding optimal filter for a given application.

II. Fitness Function

Because of statistical nature of filter training, aperture filters are in general designed to be application specific. The window shape itself, as well as any other parameter, can be chosen to exploit specific signal characteristics to improve filtering performance for given application. For instance cross-shaped window is usually better suited for filtering images with a large number of horizontal and vertical edges than rectangular window, even if both windows contain the same number of pixels, hence having equal sizes of the search spaces.

In light of the above, filter structure optimization should also be application specific, i.e. the fitness function needs to take into account filter performance over a given application specific test set. In the presented research the fitness function was constructed as the mean square error (MSE) measure between original (clean) and corrupted (noisy) image filtered by evaluated aperture.

In numerical experiments in the following part of the paper, an example application for blur removal was selected, since blur removal is a difficult task to be modeled statistically, and therefore provides good quantification of the aperture filter's performance limitations.

The analysis of constraint impact on filter performance was tested with the set of 100 greyscale pictures (8 bit) of Glasgow suburbs in 640×480 resolution was blurred with Gaussian low-pass filter with 3×3 kernel and standard deviation of 3. Gaussian filter kernel values are presented in fig. 3. From this dataset, 20 pairs were selected for testing purposes. The remaining 80 image pairs were used for training.

0.1070	0.1131	0.1070
0.1131	0.1196	0.1131
0.1070	0.1131	0.1070

Figure 3. Kernel of filter used for blurring

The domain of the fitness functions is the space of all possible structures of aperture filter within an aforementioned aperture filter definition. With the use of such definition, fitness function does not have precisely defined dimensionality. In addition, filter definition consists of both discreet and continuous parameters. As a consequence, the well-known deterministic hill-climbing optimization algorithms were not used in the research, since their use would require the loss of generality.

```
<filter>
     <position id="1" op="COPY">
          <mask width="1" height="1" center-x="0" center-y="0">
               <r>x</r>
          </mask>
     </position>
     <position id="2" op="MEDIAN">
          <mask width="3" height="3" center-x="1" center-y="1">
               <r>xxx</r>
               <r>xxx</r>
               <r>xxx</r>
          </mask>
     </position>
...
     <aperture position="1" output-ranges="1">
          <mapping op="MEAN" scale="9.7" clip-min="-9" clip-max="-3">
               <mask width="3" height="3" center-x="1" center-y="1">
                    <r>.x.</r>
                    <r>xxx</r>
                    <r>.x.</r>
               </mask>
          </mapping>
          <mask width="5" height="5" center-x="2" center-y="2">
               <r>..0..</r>
               <r>..1..</r>
               <r>01110</r>
               <r>..1..</r>
               <r>..0..</r>
          </mask>
          <param-set id="0" scale="13.6" clip-min="-6" clip-max="-3"/>
          <param-set id="1" scale="44.8" clip-min="6" clip-max="19"/>
     </aperture>
     <aperture position="2" output-ranges="10" scale="9" clip-min="-1" clip-max="1">
          <mask width="3" height="3" center-x="1" center-y="1">
               <r>x.x</r>
               <r>.x.</r>
               <r>x.x</r>
          </mask>
     </aperture>
     <fitness>123.4567</fitness>
</filter>
```

Figure 4. Example XML filter description

Different filter definitions would consist of different number of parameters, and representing them in form of a vector with *a priori* determined size would pose a threat of overly constraining filter representation to find sufficient solution for a given application. The first necessary step in aperture filter optimization was to implement flexible filter definition to describe such a structure. The XML was chosen as it is naturally suited for hierarchical object description, it is human readable and supported by a number of programming libraries. Example of XML file describing aperture filter structure is presented in listing (Fig. 4).

In general the proposed XML representation of aperture filter structure consists of the following elements (tags):

- *<filter>* – main tag containing a definition of the bundle of single aperture filters, allowing to implement pyramidal approach, for instance a bunch of filters with gradually increasing scale factor.
- *<position>* – tag defining aperture positioning function, i.e. determining the pixel grayscale value on which the aperture will be positioned.
- *<aperture>* – tag defining aperture.
    - *output-ranges* – attribute defines the number of possible output values to be estimated for each input pattern.
    - *scale* – attributes define globally the scaling factor by which input pixel values are divided
    - *clip-min* – minimal range to which the scaled values are clipped
    - *clip-max* – minimal range to which the scaled values are clipped
- *<mask>* – tag defining window shape (Fig. 5)
    - *width* – the width of the input mask
    - *height* – the height of the input mask
    - *center-x* – x coordinates of window origin
    - *center-y* – y coordinates of window origin
    - "." – pixel is excluded from the pattern
    - "x" – pixel is included from the pattern
    - 1, 2, 3 – pixels inside single mask, which will be explained later
- *<param-set>* – optional tag which allows defining a class of pixels in the input mask.
- *<mapping>* – optional tag defining a mapping of several pixels.
- *<fitness>* – tag consisting of calculated fitness value. It is used for information and performance reasons only.

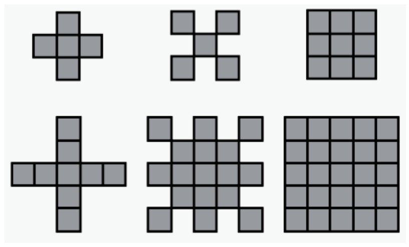
Figure 5. Example window shapes

III. Optimization algorithms

In this research the choice of optimization algorithm was determined by the definition of fitness function. Since there is no algebraic description of the function, and it's domain cannot be represented by a vector, due to hierarchical parameter structure and mixed continuous/discreet values, it is not possible to use deterministic nor hybrid algorithms.

From among the stochastic optimisation algorithms tested, Genetic Algorithm was found to be the best suited for the problems domain and the way the fitness function is found. In the research Simulated Annealing was also tested, but with worse results.

A. Genetic Algorithms

Genetic Algorithms are the very popular and wide used group of stochastic algorithms [8]. The first work on evolution processes simulations were done as early as 1950s, however more attention of the researchers was drawn into the subject starting from 1980s. The optimization is inspired by the evolution processes observed in nature, which transformed into mathematical models proved to be very well suited for many optimization tasks.

The main problem with GA is considerate computational cost (resulting from testing a significant number of possible solutions), which in case of some applications might be too high, rendering the method ineffective for the task. Other common problems are the convergence issues near the global minimum, which requires more sophisticated conditions to effectively stop the optimization.

Evolution algorithm is a iterative process of adapting a population of potential solutions by means of evolutionary operators, such as mutation, selection, crossover and inheritance. As an outcome of each iteration (often called epoch) a new populations of potential solutions is generated, usually better then the previous.

Implemented algorithm is a classic version of Genetic Algorithm with elitism (i.e. the protection of the best-fit individual from its parameter deterioration by genetic operators).

In the presented research, the crossover operator exchanges whole aperture definitions between with the use on one-point approach. The split point is randomly selected, however with the restriction that the overall number of apertures inside a single filter would not exceed 10, as this could lead to uncontrolled growth of the filter structure, whereas empirical experience shows that increasing the number of apertures over ten is unlikely to further improve filtering quality.

In the proposed solution, the mutation is performed in the following steps:

1. random choice of aperture to be modified
2. random choice of one of the operation:
   - inserting new pixel into the mask at random position
   - deleting pixel from the mask from random position
   - changing one of the scaling and clipping parameters of randomly selected pixel. The exact parameter to be changed is also randomly selected, and its value is changed by random value from 10% to 10% of original value

The most difficult task in Genetic Algorithm implementation is the design of a selection operator.

In the presented research, roulette selection strategy was chosen, and the selected population undergoes crossover and cloning (rarely) operations followed by random mutations, with the succession of the fittest individual.

Main disadvantage of this method is the gradual weakening of evolutionary pressure with the progress of evolution process. With a population of similarly fit individuals, the selection is actually random.

B. *Simulated Annealing*

This algorithm is an adaptation of the annealing process known from metallurgy for optimization tasks proposed independently by S. Kirkpatrick in 1983 [9] and V. Cerny in 1985 [10]. The annealing is performed by heating the metal above the melting point, and next slowly cooling it down, during which the atoms are gradually distributed uniformly into the crystal structure characterized by the lowest inner energy. The atoms change places in the structure with the probability dependent on the energy level in both new and old place as well as the temperature.

In Simulated Annealing it is assumed that the equivalent of energy is the provided fitness function, and the temperature is controlled by a certain cooling scheme. The optimization follows by iterative change of state of the solution (switching from one possible solution to another), with the temperature degrading by some factor on every iteration.

In the research a typical Boltzmann cooling schedule was implemented. The shift to a state with higher energy is made only if the equation 1 is satisfied:

$$\exp\left(-\frac{\Delta E}{T}\right) > r_{[0,1]} \qquad (1)$$

where ΔE is the energy change between the new and old state, T is current temperature given by eq. 2 and $r_{[0,1]}$ is random value in range [0,1].

$$T(i) \cong \frac{1}{\ln(i)} \qquad (2)$$

Main disadvantage of this approach is the necessity to provide the cooling schedule function. There are no clear criteria allowing for proper selection of this function. Obviously, if the process is cooled too fast it could stuck in local minimum, whereas cooling too slow would lead to almost pure random search, and the algorithm would not converge into acceptable solution in reasonable time. In practice finding proper cooling schedule can be difficult.

The temperature in the implementation was changing in accordance to the logarithmic cooling schedule, which apparently was not the best choice for this application. However due to the computational complexity of the process, finding a better cooling scheme was abandoned in favor of less sensitive GA approach, described above.

IV. Results

In order to test the usefulness of the implemented optimization algorithms, two numerical experiments were prepared. In the first the best aperture filter found in the previous experiments was further optimized in order to show, that the best human designed filter can still usually be improved further for a given application by the means of presented optimization methods. In the other experiment the optimized filter was generated randomly. The aim of the experiment was to check if the automatically designed (by random filter optimization) filter is comparable with the one designed by human.

The optimization process was presented in diagrams plotting fitness value of the best found solution against the iteration number.

In Fig. 6 an example XML definition of single aperture before and after optimisation process is presented. In Fig. 7 the diagram of the optimisation process by Simulated Annealing is presented. In Fig. 8 the diagram of the optimisation process by Genetic Algorithm is presented.

```
<aperture position="1" output-ranges="1" scale="8" clip-min="-2" clip-max="2">
    <mask width="5" height="5" center-x="2" center-y="2">
        <r>..x..</r>
        <r>..x..</r>
        <r>xx.xx</r>
        <r>..x..</r>
        <r>..x..</r>
    </mask>
</aperture>
```

```
<aperture position="1" output-ranges="1" scale="8" clip-min="-2" clip-max="2">
    <mask width="5" height="5" center-x="2" center-y="2">
        <r>..x..</r>
        <r>..0..</r>
        <r>x1.2x</r>
        <r>..1..</r>
        <r>..x..</r>
    </mask>
    <param-set id="0" scale="7.4" clip-min="-2" clip-max="2"/>
    <param-set id="1" scale="8" clip-min="-2" clip-max="1"/>
    <param-set id="2" scale="8" clip-min="-1" clip-max="2"/>
</aperture>
```

Figure 6. Example of optimization for single aperture

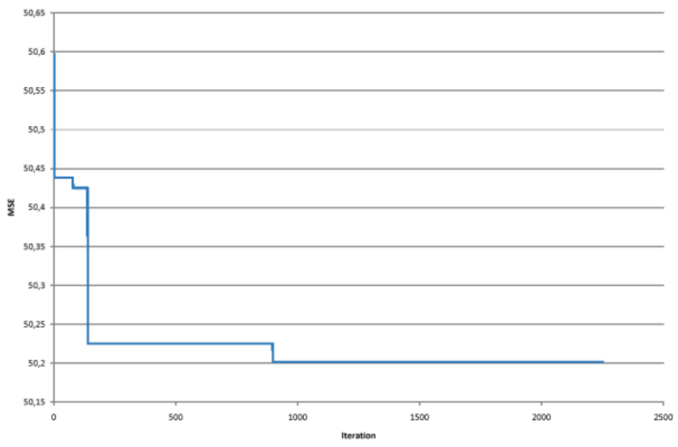

Figure 7. Results of filter structure optimization by Simulated Annealing

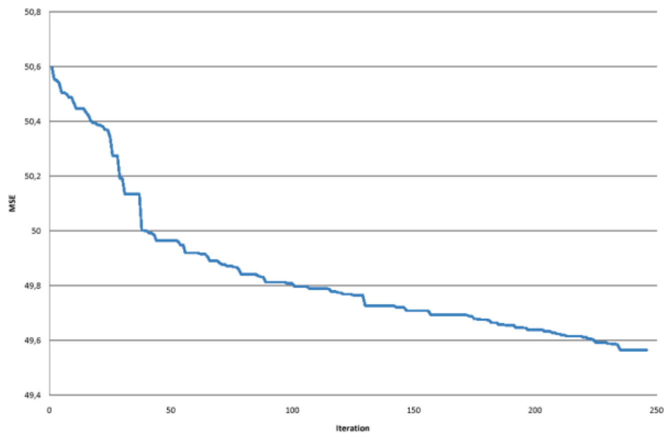

Figure 8. Results of filter structure optimization by Genetic Algorithm (experiment 1)

It can be seen that in case of Simulated Annealing the optimization process was getting stuck for a significant number of iterations in local minima, unable to shift out of them. The reason for this was poorly selected cooling schedule, with the temperature settling down too fast, which severely impaired the ability of the system to escape from local minimum. Another test runs were tried, with different cooling schedules, however the results were unsatisfactory, and finally this approach was abandoned in favor of Genetic Optimization, which turned out to be much less sensitive for process adjustments.

In Fig. 9 the diagram of the optimization process is presented.

It can be seen that optimization improved filter performance by a significant factor. Filtering results are presented in fig. 10. Images filtered with optimized filter are visually better than the blurred ones, as well as the images filtered by the initial random filter.

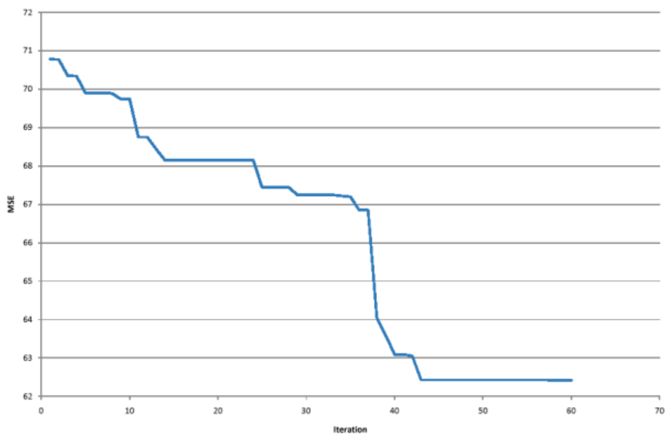

Figure 9. Results of filter structure optimization by Genetic Algorithm (Experiment 2)

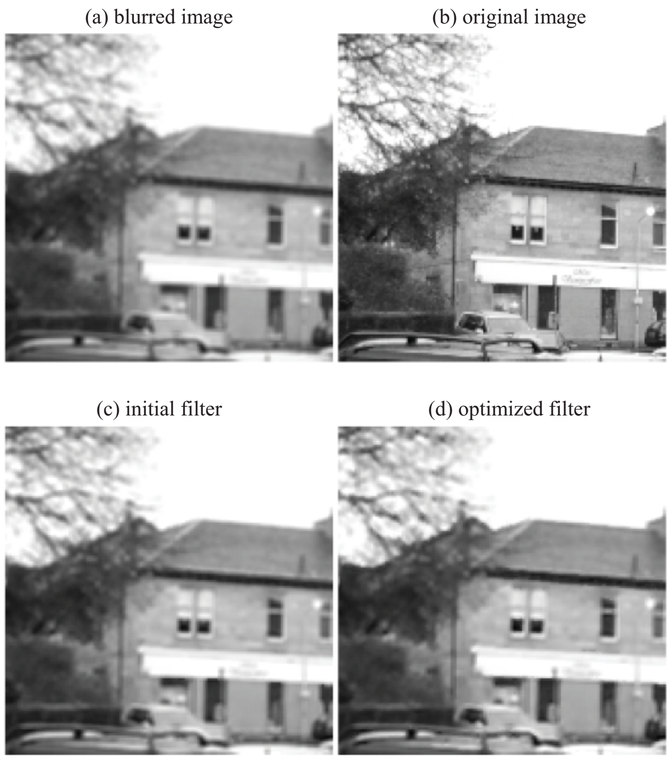

Figure 10. Image fragment filtered by initial random and optimized filters

V. Summary and Conclusions

Stochastic optimization methods, namely Genetic Algorithm can be used to automatically improve aperture filter parameters for a given application. Both randomly generated, as well as human designed filters can be tuned by the proposed method to improve filtering quality. It is however important to note, that the optimization process itself is computationally expensive, and the resulting improvement is not necessarily sufficient for a given task. From the two tested optimization methods, Genetic Algorithm in general performed better, as it is less sensitive for user controllable parameters.

Aperture filters are a relatively new class of image operators, and as such require more research for further improvements. Authors researched two different approaches. The first approach was to improve on the filter definition itself, providing a way to balance the constraint cost against the estimation cost for a given application [7]. The second approach was to implement an optimisation framework in order to find a better suited filter structure. Both approaches were verified in the research to be fruitful.

The optimization process presented in this paper is general and does not depend on any particular application. Presented deblurring application was only an example of aperture possibilities. In the further research we will explore a more practical usage of presented aperture filter design methodology.

Acknowledgment

Jakub Chłapiński thanks PhD. Marek Kamiński from the Department of Microelectronics and Computer Science, Technical University of Lodz.

References

[1] Roberto Hirata, Edward R. Dougherty, and Junior Barrera. Aperture filters. Signal Processing, 80:697–721, 2000.

[2] Roberto Hirata, Marcel Brun, Junior Barrera, and Edward R. Dougherty. Multiresolution design of aperture operators. Mathematical Imaging and Vision, 16:199–222, 2002.

[3] Roberto Hirata, Marcel Brun, Junior Barrera, and Edward R. Dougherty. Multiresolution design of aperture operators. Mathematical Imaging and Vision, 16:199–222, 2002.

[4] Alan C. Green, Stephen Marshall, David Greenhalgh, and Edward R. Dougherty. Design of multi-mask aperture filters. Signal Processing, 83:1961–1971, 2003.

[5] Roberto Hirata, Edward R. Dougherty, and Junior Barrera. Design of greyscale nonlinear filters via multiresolution apertures. In EUSIPCO 2000, Tampere, Finland, September 2000.

[6] Edward R. Dougherty and Divyendu Sinha. Computational mathematical morphology. Signal Processing, 38:21–29, 1994.

[7] Jakub Chłapiński, Stephen Marshall. Releasing Aperture Filter Constraints, EUSIPCO 2007, September 2007, Poznan, Poland.

[8] John R. Koza and J. Rice. Genetic Programming on the Programming by means of Natural Selection. The MIT Press, Cambridge, 1992.

[9] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. Science, New Series 220 (4598):671–680, 1983.

[10] V. Cerny. A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. Journal of Optimization Theory and Applications, 45:41–51, 1985.